Semi-Implicit Update

1 Derivation

At this point in the simulation, we want to update the grid node velocity. Explicitly, this is just

$$\boldsymbol{v}_{\boldsymbol{i}}^{n+1} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \frac{\partial \boldsymbol{v}_{\boldsymbol{i}}^{n}}{\partial n} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \Delta t \frac{\partial \boldsymbol{v}_{\boldsymbol{i}}^{n}}{\partial t} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \Delta t \frac{\boldsymbol{f}_{\boldsymbol{i}}^{n}}{m_{\boldsymbol{i}}}, \tag{1}$$

by Newton's second law. The backward Euler method can be used to express the new velocity implicitly:

$$\boldsymbol{v}_{\boldsymbol{i}}^{n+1} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \frac{\partial \boldsymbol{v}_{\boldsymbol{i}}^{n+1}}{\partial n} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \Delta t \frac{\partial \boldsymbol{v}_{\boldsymbol{i}}^{n+1}}{\partial t} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \Delta t \frac{\boldsymbol{f}_{\boldsymbol{i}}^{n+1}}{m_{\boldsymbol{i}}}. \tag{2}$$

The "semi-implicit" update employed by the authors essentially interpolates between these two schemes:

$$\begin{aligned}
\boldsymbol{v}_{\boldsymbol{i}}^{n+1} &= \boldsymbol{v}_{\boldsymbol{i}}^{n} + (1-\beta)\Delta t \frac{\boldsymbol{f}_{\boldsymbol{i}}^{n}}{m_{\boldsymbol{i}}} + \beta \Delta t \frac{\boldsymbol{f}_{\boldsymbol{i}}^{n+1}}{m_{\boldsymbol{i}}} \\
&= \boldsymbol{v}_{\boldsymbol{i}}^{n} + \frac{\Delta t}{m_{\boldsymbol{i}}}\Big(\boldsymbol{f}_{\boldsymbol{i}}^{n} + \beta \left(\boldsymbol{f}_{\boldsymbol{i}}^{n+1} - \boldsymbol{f}_{\boldsymbol{i}}^{n} \right) \Big) \\
&\approx \boldsymbol{v}_{\boldsymbol{i}}^{n} + \frac{\Delta t}{m_{\boldsymbol{i}}}\Big(\boldsymbol{f}_{\boldsymbol{i}}^{n} + \beta \Delta t \frac{\partial \boldsymbol{f}_{\boldsymbol{i}}^{n}}{\partial t} \Big).
\end{aligned}$$

Note that

$$\frac{\partial \boldsymbol{f}_{\boldsymbol{i}}^{n}}{\partial t} \approx \sum_{\boldsymbol{j}} \frac{\partial \boldsymbol{f}_{\boldsymbol{i}}^{n}}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}} \boldsymbol{v}_{\boldsymbol{j}}^{n+1},$$

where $\hat{\boldsymbol{x}}_{\boldsymbol{i}} = \boldsymbol{x}_{\boldsymbol{i}} + \Delta t\, \boldsymbol{v}_{\boldsymbol{i}}$. So, the update scheme is

$$\boldsymbol{v}_{\boldsymbol{i}}^{n+1} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \frac{\Delta t}{m_{\boldsymbol{i}}} \left(\boldsymbol{f}_{\boldsymbol{i}}^{n} + \beta \Delta t \sum_{\boldsymbol{j}} \frac{\partial \boldsymbol{f}_{\boldsymbol{i}}^{n}}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}} \boldsymbol{v}_{\boldsymbol{j}}^{n+1} \right). \tag{3}$$

You can see that when $\beta = 0$, this update is equivalent to the explicit update, and when $\beta = 1$, this update is (approximately) equivalent to the backward Euler update.

Define the explicitly updated velocity as

$$\boldsymbol{v}_{\boldsymbol{i}}^{*} = \boldsymbol{v}_{\boldsymbol{i}}^{n} + \Delta t \frac{\boldsymbol{f}_{\boldsymbol{i}}^{n}}{m_{\boldsymbol{i}}} \tag{4}$$

By definition,

$$\frac{\partial^2 \Phi^n}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{i}} \partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}} = -\frac{\partial \boldsymbol{f}_{\boldsymbol{i}}^{n}}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}}. \tag{5}$$

Using (3), (4), and (5), we have that

$$\boldsymbol{v}_{\boldsymbol{i}}^{*} = \boldsymbol{v}_{\boldsymbol{i}}^{n+1} + \frac{\beta \Delta t^2}{m_{\boldsymbol{i}}} \sum_{\boldsymbol{j}} \frac{\partial^2 \Phi^n}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{i}} \partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}} \boldsymbol{v}_{\boldsymbol{j}}^{n+1} = \sum_{\boldsymbol{j}} \left(\boldsymbol{I} \delta_{\boldsymbol{ij}} + \frac{\beta \Delta t^2}{m_{\boldsymbol{i}}} \frac{\partial^2 \Phi^n}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{i}} \partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}} \right) \boldsymbol{v}_{\boldsymbol{j}}^{n+1}. \tag{6}$$

2 Implementation

This update can be formulated as a large system of linear equations. Define $\boldsymbol{V}$ as the vector of all the updated grid node velocities, such that

$$\boldsymbol{V} \triangleq \left(\boldsymbol{v}_{1x}^{n+1},\ \boldsymbol{v}_{1y}^{n+1},\ \boldsymbol{v}_{1z}^{n+1},\ \ldots\ ,\ \boldsymbol{v}_{gx}^{n+1},\ \boldsymbol{v}_{gy}^{n+1},\ \boldsymbol{v}_{gz}^{n+1}\right)^T,$$

where g is the total number of grid nodes. Likewise, define $\boldsymbol{V}^*$ as the vector of all the explicitly update grid node velocities, such that

$$\boldsymbol{V}^* \triangleq \left(\boldsymbol{v}_{1x}^{*},\ \boldsymbol{v}_{1y}^{*},\ \boldsymbol{v}_{1z}^{*},\ \ldots\ ,\ \boldsymbol{v}_{gx}^{*},\ \boldsymbol{v}_{gy}^{*},\ \boldsymbol{v}_{gz}^{*}\right)^T.$$

We want to express (6) in matrix form as $\boldsymbol{V}^* = \boldsymbol{E}\boldsymbol{V}$. Both $\boldsymbol{V}$ and $\boldsymbol{V}^*$ have dimensions $3g \times 1$, so we want to compute $\boldsymbol{E}$, a matrix of dimensions $3g \times 3g$.

We can think of $\boldsymbol{E}$ as a $g \times g$ matrix of 3×3 sub-matrices. So how do we compute these sub-matrices? The key term in (6) is the Hessian of the potential. The "action" of this Hessian on an arbitrary increment $\delta \boldsymbol{u}$ is

$$\delta \boldsymbol{f_i} = -\sum_{\boldsymbol{j}} \frac{\partial^2 \Phi}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{i}} \partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}} \delta \boldsymbol{u_j} = -\sum_p V_p^0 \boldsymbol{A}_p (\boldsymbol{F}_{Ep}^n)^T \nabla w_{\boldsymbol{j}p}^n, \tag{7}$$

where

$$\boldsymbol{A}_p = \frac{\partial^2 \Psi}{\partial \boldsymbol{F}_{Ep} \partial \boldsymbol{F}_{Ep}} : \left(\sum_{\boldsymbol{j}} \delta \boldsymbol{u_j} (\nabla w_{\boldsymbol{j}p})^T \boldsymbol{F}_{Ep}^n\right). \tag{8}$$

Generally, the notation $a = \boldsymbol{B} : \boldsymbol{C}$ indicates the Frobenius inner product of matrices B and C, where

$$a = \boldsymbol{B} : \boldsymbol{C} = \sum_i \sum_j \boldsymbol{B}_{ij} \boldsymbol{C}_{ij}.$$

The authors extend this notation to the form seen in (8), $\boldsymbol{A} = \boldsymbol{B} : \boldsymbol{C}$ to indicate that

$$\boldsymbol{A}_{ij} = \boldsymbol{B}_{ij} : \boldsymbol{C}. \tag{9}$$

In other words, $\boldsymbol{B}$ is some matrix of sub-matrices, such that $\boldsymbol{B}_{ij}$ itself is a matrix. Each entry of $\boldsymbol{A}$ is the Frobenius inner product of the corresponding sub-matrix in $\boldsymbol{B}$ with $\boldsymbol{C}$. In the case of (8), $\boldsymbol{A}$ and $\boldsymbol{C}$ are 3×3 matrices while $\boldsymbol{B}$ is a 3×3 matrix of 3×3 sub-matrices.

For brevity's sake, let's define

$$\delta \boldsymbol{F}_{Ep} \triangleq \sum_{\boldsymbol{j}} \delta \boldsymbol{u_j} (\nabla w_{\boldsymbol{j}p})^T \boldsymbol{F}_{Ep}^n,$$

such that

$$\boldsymbol{A}_p = \frac{\partial^2 \Psi}{\partial \boldsymbol{F}_{Ep} \partial \boldsymbol{F}_{Ep}} : \delta \boldsymbol{F}_{Ep}.$$

In the technical report, the authors show that

$$\frac{\partial^2 \Psi}{\partial \boldsymbol{F}_{Ep} \partial \boldsymbol{F}_{Ep}} : \delta \boldsymbol{F}_{Ep} = 2\mu\, \delta \boldsymbol{F}_{Ep} - 2\mu\, \delta \boldsymbol{R}_{Ep} + \lambda J_{Ep} \boldsymbol{F}_{Ep}^{-T} (J_{Ep} \boldsymbol{F}_{Ep}^{-T} : \delta \boldsymbol{F}_{Ep}) + \lambda (J_{Ep} - 1)\, \delta (J_{Ep} \boldsymbol{F}_{Ep}^{-T}), \tag{10}$$

where as usual, $\boldsymbol{F}_{Ep} = \boldsymbol{R}_{Ep} \boldsymbol{S}_{Ep}$ by polar decomposition, and $J_{Ep} = \det(\boldsymbol{F}_{Ep})$.